Modeling and Simulation to the Design of ΣΔ Fractional-N Frequency Synthesizer

Shuilong Huang[1], Huainan Ma[2], and Zhihua Wang[1]
[1]Department of Electronics Engineering, Tsinghua University, Beijing 100084, China
[2]RFIC Corporation, Beijing 100084, China

Abstract

A set of behavioral voltage-domain verilogA/verilog models allowing a systematic design of the ΣΔ fractional-N frequency synthesizer is discussed in the paper. The approach allows the designer to accurately predict the dynamic or stable characteristic of the closed loop by including nonlinear effects of building blocks in the models. The proposed models are implemented in a three-order ΣΔ fractional-N PLL based frequency synthesizer with a 60MHz frequency tuning range. Cadence SpectreVerilog simulation results show that behavioral modeling can provide a great speed-up over circuit-level simulation. Synchronously, the phase noise, spurs and settling time can also be accurately predicted, so it is helpful to a grasp of the fundamentals at the early stage of the design and optimization design at the system level. The key simulation results have been compared against measured results obtained from an actual prototype validating the effectiveness of the proposed models.

1 Introduction

Fractional-N PLL based frequency synthesizer is widely used in modern wireless communication system due to high frequency resolution and fast settling time. However, it also encounters some challenges. First, factional-N PLL frequency synthesizer is a complicated mixed-signal system, and there is no stable periodic solution, and even no tool can effectively and quickly simulate the closed fractional-N PLL frequency synthesizer, especially for transistor level circuit. In addition, the time constants of different loop components may vary over a very wide range, so that the total simulation time must be at least one order of magnitude greater than the largest time constant of the loop filter for the loop to be locked. If phase noise needs to be studied, an even greater simulation time is required to obtain a sufficiently large sample space to accurately predict the random behavior. This would lead to high cost and slow turnaround in design.

Much effort has been developed to solve the above problem. In [1], Behavioral modeling with Matlab/C can acquire system simulation result, but it makes system design be independent to component circuit, and optimization can not be realized in system level, so it is not suitable for top-down design. Macromodel [2] is used for modeling the digital part. The simulation speed is quickened, whereas the phase noise/spur analysis is yet a problem. In [3], phase-domain and voltage-domain model can effectively predict the phase noise or jitter. However, the models do not include the nonlinear effects like charge pump mismatch or output delay difference of phase-frequency detector (PFD) etc., so it can not predict spur performance. In addition, the voltage-domain models can not be implemented in the fractional-N architecture. The behavioral models [4] consider the effects of charge pump mismatch and output delay difference of PFD, but the jitter of component circuit is not included in the model.

The primary contribution of the paper is to demonstrate a behavioral simulation methodology and a set of models using verilogA/verilog description. Particular attention is devoted to model the nonideal characteristics of the building blocks. The circuit-level simulation results of the individual building blocks can be back-annotated onto the behavioral models for further verification. The models can be used to accurately predict the phase noise/spur performance, and the concepts introduced can be applied to a number of PLL applications, which is especially applied in the design of the ΣΔ fractional-N frequency synthesizer.

The paper is organized as follows. Section 2 shows the mixed-signal simulation flow and strategy. Section 3 presents the loop component models based on verilogA/verilog language. Finally, Simulation results are shown in Section 4, and conclusions are given in Section 5.

2 Mixed-signal simulation flow and strategy

The ΣΔ fractional-N frequency synthesizer modeled in the paper is shown in Fig. 1, which can track phase/frequency of input signal and output a series of high spectral purity frequencies. The architecture consists of a reference oscillator (OSC), a frequency doubler, a phase/frequency

978-3-9810801-2-4/DATE07 © 2007 EDAA

detector (PFD), a charge pump (CP), a loop filter (LF), a voltage-controlled oscillator (VCO), a ΣΔ modulator, and a divider. The frequency doubler is used to extend the reference frequency and the PFD is used to detect the difference between the input reference frequency and the output frequency of VCO. It exports the voltage signal proportional to this difference, then drives the charge pump circuit. The output current of charge pump is converted to voltage by the loop filter to control the output frequency of the VCO. The ΣΔ modulator is used to generate a random sequence with the desired duty cycle to control the divider in the feedback path.

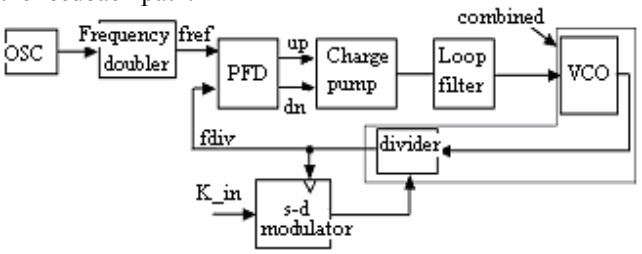


Fig. 1: ΣΔ fractional-N frequency synthesizer.

The proposed mixed-signal simulation flow is shown in Fig. 2. The analog loop components like VCO are described with verilogA language, and the digital loop components like divider and ΣΔ modulator are modeled based on verilog language. The loop filter remains a full circuit-level model and the noise behavior of the filter is naturally included in the resistor model. Due to the lack of periodic stable solution in fractional-N architecture, PSS and PNOISE analysis fails to predict stable performance, so the mathematical tool Matlab is implemented to post-process the simulation data. The behavioral model can be replaced by transistor level-model or layout-level abstracted model on the hierarchy-editor environment, so it is suitable for top-down design flow.

To be an effective behavioral model, some strategies are needed to be considered. First, the parameters that can represent the characteristics of circuit should be contained as many as possible. Second, nonlinear effects should be included in the behavioral model, which influence the spurs and jitter of the closed loop. The spur is a deterministic signal. The amount of which mainly depends on current mismatch and leakage current of charge pump, output delay difference of PFD, linearity of PFD/CP, and output of ΣΔ modulator. The design and optimization is easily realized at system-level and circuit-level design. However, Jitter is an uncertainty or randomness of signal timing, which can be reformulated as a phase noise. Accurate analysis and prediction of jitter is very difficult, so just simple synchronous and accumulating jitter is considered here to easily estimate jitter behavior of the PLL. Third, it is desirable to combine jitter sources to the degree possible. For instance, the output-referred noise of divider and the input-referred noise of the PFD/CP are combined with the output noise of OSC. The models after combination will run more efficiently by removing support for jitter. Last, to quicken simulation speed, it is possible to include the frequency division ratio of divider as part of the VCO by adjusting the VCO gain and jitter. If the division ratio of divider is large, the simulation runs much faster because the high VCO output frequency is never generated. But the simple mergence of the two blocks is just suitable for integer-N frequency synthesizer. In the fraction-N architecture, the transient ratio is not constant, so the mergence should consider the effects of ΣΔ modulator.

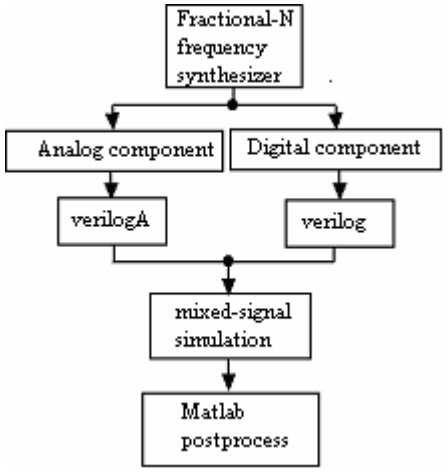


Fig. 2: Mixed-signal simulation flow.

3 Circuit model based on verilogA/Verilog

3.1 OSC Model

The model is given in Fig.3, which supports two jitter parameters. The *accJitter* parameter is used to model the accumulating jitter of the OSC, and the *syncJitter* parameter is used to model the synchronous jitter of divider, PFD, and charge pump. The *duty* parameter is used to model the duty cycle ratio of the OSC signal. The timer() function is used to model accumulating or synchronous jitter of OSC. At every output transition, the next transition is scheduled using the timer () function to be $T/K + J\delta/\sqrt{K}$ in the future, where δ is the unit-variance zero-mean random process and K is the number of output transitions per period, typically, K=2.

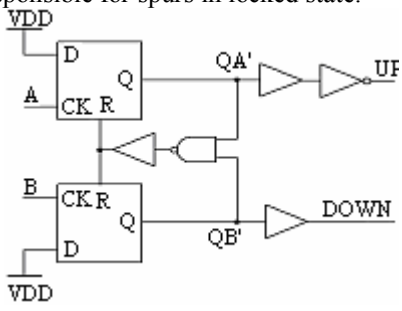
```
`include "constants.vams"
`include "disciplines.vams"
module OSC_behav(out);
output out; electrical out;
parameter real freq=19.2M from(0:inf);
parameter real Vlo=0,Vhi=2.7;
parameter real tt=0.1n;
parameter real accJitter=30f from[0:0.1/freq); parameter real
syncJitter=400f from[0:0.1*ratio/freq);
```

```
integer n,accSeed,syncSeed;
accSD=accJitter*sqrt(ratio/2);
 syncSD=syncJitter;
  next1=(1-duty)/freq+$abstime;
  next2=duty/freq+$abstime;end
 @(timer(next1+dt))begin
 n=!n;
dT=accSD*$dist_normal(accSeed,0,1);
dt=syncSD*$dist_normal(syncSeed,0,1);
next1=next2+(1-duty) /freq+dT; end
@(timer(next2+dt))begin
 n=!n;
dT=accSD*$dist_normal(accSeed,0,1);
dt=syncSD*$dist_normal(syncSeed,0,1);
next2=next1+duty/freq+dT; end
V(out)<+transition(n?Vhi:Vlo,0,tt);
end  endmodule
```

Fig. 3: OSC model.

3.2 PFD Model

The PFD shown in Fig. 4(a) employs sequential logic to create three states and responds to the rising edges of the two inputs. When reference signal ref leads feedback signal fbk, the output UP remains high until fbk goes high, at which point UP returns to zero. The behavior is similar for the fbk input. Fig.4(b) illustrates the modeling approach for the PFD. @cross statement is triggered when the input crosses the threshold in the user specified direction, used to model two D triggers shown in Fig. 4(a). On every transition of the fbk or ref input in direction *dir*, the output is set to "1". The outputs of the two D triggers are resetted to "0" on every transition of reset input in the direction *dir*. The transition () function is used to model reset path delay and output delay difference. The *td1* parameter represents the reset path delay, which is relative to the dead band of the charge pump. The *td2* parameter and *td3* parameter are the delay of output path, which is responsible for spurs in locked state.



(a)Circuit structure of PFD

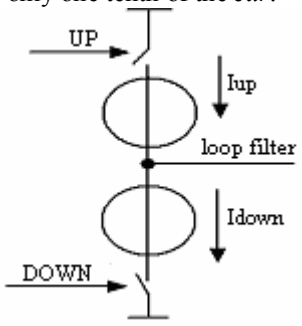
```
`include "constants.vams"
`include "disciplines.vams"
module pfd_behav(ref,fbk,up,down);
input ref, fbk;
output up,down;
electrical ref,fbk,up,down;
integer resetx,reset,ax,bx;
parameter integer dir=1 from[-1:1] exclude 0;
parameter real tt=0.01n from(0:inf);
parameter real td1=0.10n from(0:inf);
parameter real td2=0.1n from(0:inf);
parameter real td3=0.4n from(0:inf);
parameter real ttol=1p from(0:inf);
parameter real Vlo=0,Vhi=2.5;
analog begin
 @(cross(V(ref)-0.5,dir,ttol) or cross(reset-0.5,dir,ttol))
begin
    if(reset==0) ax=1;  else  ax=0;   end
  @(cross(V(fbk)-0.5,dir,ttol) or cross(reset-0.5,dir,ttol))
begin
    if(reset==0) bx=1; else    bx=0;   end
  if(ax==1 && bx==1)resetx=1;   else resetx=0;
 reset=transition(resetx,td1,0.1n,0.1n);
  V(up)<+transition(ax ? Vlo : Vhi,td2,0.1n,0.1n);
  V(down)<+transition(bx ? Vhi:Vlo,td3,0.1n,0.1n);
end
endmodule
```

(b) Model of PFD

Fig. 4: Circuit and model of PFD.

3.3 Charge Pump Model

Charge pump in Fig. 5(a) pumps current into the loop filter. The action is controlled by the UP and DOWN signals from the PFD. When the UP signal is low, Iup is pump into the filter and results in an increase in the voltage to VCO. Similarly, when the DOWN signal is high, the current Idown discharges from the filter resulting in a decrease in the voltage to the VCO. The corresponding model is shown in Fig. 5(b). For a simple model, just up/down current mismatch and dead band of charge pump are considered in the proposed model. The former is responsible for reference spurs, and the latter is caused by the very narrow pulse of UP or DOWN input, which is relative for phase noise. The other nonlinearities like clock feedthrough, charge sharing, charge injection, clock injection can be equivalent to the amount of current mismatch or dead band. The *td* parameter is used to model the dead band. The *Iextra* parameter is used to model the current mismatching characteristic of charge pump. If the control voltage on the loop filter is out of the saturation range of charge pump, the output current of the current source is only one tenth of the *cur*.



(a) Circuit structure of charge pump

```
`include "constants.vams"
`include "disciplines.vams"
module chargepumpnew_behav(up,dn,Iout);
input up,dn;
electrical up,dn;
electrical Iout;
parameter real cur=200u;
parameter real tt=0.1n from(0:inf);
parameter real td=1.20n from(0:inf);
real out, upx,dnx, cur_mid;
analog begin
  upx=V(up);   dnx=V(dn);
   if(V(Iout)>=0.4 && V(Iout)<=1.4) cur_mid=cur;
   else                             cur_mid=cur/10;
   if(upx<=0.50 && dnx<=0.5)   out=-cur_mid;
   else if(upx>=2.2 && dnx>=2.2)  out=cur_mid+6u;
   else if(upx<=0.5 && dnx>=2.2)  out=cur_mid/100;
   else  out=cur_mid/1000;
 I(Iout)<+transition(out,td,tt);
end
endmodule
```

(b) model of charge pump

Fig. 5: Circuit and model charge pump.

3.4 VCO/Divider Model

Due to high output frequency of VCO in the radio applications, it is very time-consuming to simulate the closed-loop PLL frequency synthesizer at the circuit-level and even the behavioral-level, so it is very necessary to include the frequency division ratio of the divider as part of the VCO to quicken the simulation speed. However, because the transient division ratio is not fixed in the ΣΔ modulator fractional-N architecture, the divider model can not be simply included in the VCO model. A solution is proposed in the paper. The output of VCO is divided into two parts. One is to provide divided signal to PFD for comparing, and the division ratio is the transient modulated result by the ΣΔ modulator. The other part is used to analyze the spectral purity and the division ratio is the fixed averaging value. Fig. 6 shows the model description of VCO/divider. It is mainly constructed by using the three serial operations. First the input signal is scaled to compute the desired output frequency combining with the output of ΣΔ modulator (f1-f5). Then the frequency is integrated to compute the output phase. Finally, the phase is used to generate the desired output signal by computing with idtmod () function. The first two @cross statements are used to record the jitter at every clock cycle, and write the jitter data into *periods.m* file. The next two @cross statements are used to realize the divider function. The *dT* parameter is updated twice in every clock cycle, which represents the jitter of VCO and divider. The *divratio/N_div* parameter is used to model the change of the divide ratio in every clock cycle. The system function $dist_normal() is used to produce the uniform distributed random data, and the system function $abstime is used to record the current time.

```
`include "constants.vams"
`include "disciplines.vams"
module vco_div_behav(f1,f2,f3,f4,f5,vin,fout);
input vin, f1,f2,f3,f4,f5;
output fout;
electrical vin,fout, f1,f2,f3,f4,f5;
parameter integer dir=+1 from[-1:+1] exclude 0;
parameter real N=32, N_div=43.4375;
parameter real Vmin=0.5, Vmax=2.2;
parameter real Fmin=1.644G, Fmax=1.695G;
parameter real Vlo=0,Vhi=2.7;
parameter real tt=0.01/Fmax from(0:inf);
parameter real jitter=30.0f from[0:0.25/Fmax);
parameter real ttol=1u/Fmax from(0:1/Fmax);
parameter real outStart=120u from(1/Fmin:inf);
parameter real td=0.1n from(0:inf);
real freq, phase,dT,delta,prev;
integer n,seed,d1,d2,d3,d4,d5,fp1,divratio;
analog begin
@(initial_step)  begin
  seed=-561;
  prev=$abstime; delta=jitter*sqrt(2*N_div);
  fp1=$fopen("periods.m");
end
  if(V(f1)>=2.2) d1=1; else d1=0;
  if(V(f2)>=2.2) d2=1; else d2=0;
  if(V(f3)>=2.2) d3=1; else d3=0;
  if(V(f4)>=2.2) d4=1; else d4=0;
  if(V(f5)>=2.2) d5=1; else d5=0;
  divratio=N+d1+2*d2+4*d3+8*d4+16*d5;
freq=(V(vin)-Vmin)*(Fmax-Fmin)/(Vmax-Vmin)+Fmin;
if(freq>Fmax) freq=Fmax;  if(freq<Fmin)freq=Fmin;
freq=(freq/N_div)/(1+dT*freq/N_div);
phase=6.28*idtmod(freq,0.0,1.0,-0.5);
@(cross(phase+1.57,+1,ttol))
       dT=delta*$dist_normal(seed,0,1);
@(cross(phase-1.57,+1,ttol)) begin
     dT=delta*$dist_normal(seed,0,1);
 if($abstime>outStart) $fstrobe(fp1,"%0.10e",$abstime-rev);
    prev=$abstime;
     end
@(final_step) $fclose(fp1);
@(cross(phase+divratio/N_div,+1,ttol))  n=Vhi;
@(cross(phase-divratio/N_div,+1,ttol)) n=Vlo;
    V(fout)<+transition(n,td+dx,tt);
end
endmodule
```

Fig. 6: VCO/Divider model.

3.5 ΣΔ modulator & Frequency doubler Model

ΣΔ modulator [5] shown in Fig.7 is a key block in the frequency synthesizer, used to produce the fractional part of the division ratio. Due to a pure digital circuit, ΣΔ modulator is modeled based on verilog language, shown in Fig. 8 where the LSB of the input sequence is dithered to improve the noise performance. The assign statement is used to assign wire variable, whereas always@ block is used to model D trigger. Multiplying 2, 1.5 or 0.5 is realized by left-shift or right-shift operation, to reduce the layout area. The highest 4bits of the *sum7* parameter is quantized. The operation is based on complement and the MSB of the operand is the sign bit.

The frequency doubler shown in Fig.9 is a very simple digital block, which modeling process is similar to that of ΣΔ modulator, so it will not be repeated.

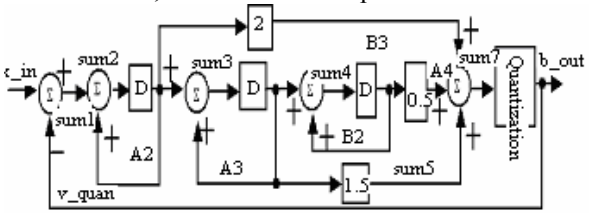


Fig. 7: Three-order ΣΔ modulator.

```
module modulator(clk, rst,dither_en, bout, k_dth);
input clk, dither_en;
input[21:0] k_dth;
output[4:0] bout;
reg[4:0] bout;
reg[3:0] bout_pre;
reg[21:0] A2, B3, A3, B2, A4;
wire[21:0] sum1,sum2,sum3,sum4,sum5,sum6,sum7,
            v_quan, v_back;
 assign sum1 ={k_dth[21:1],1'b0} + v_back;
 assign sum2 = sum1 + A2;
 assign sum3 = A2 + A3;
 assign sum4 = A3 + B2;
 assign sum6 = B3 + A4;
 assign sum7 = sum5 + sum6;
 always @(posedge clk or posedge rst) begin
    if(rst) A2 <= 22'h0;
       else   A2 <= sum2;
 end
 assign B3 ={A2[20:0],1'b0}; //B3 = 2 * A2
always @(posedge clk or posedge rst)
begin
    if(rst)     A3 <= 22'h0;      else        A3 <= sum3;
 end
 always @(posedge clk or posedge rst)
begin
    if(rst)    B2 <= 22'h0;
    else       B2 <= sum4; end
 assign A4= {B2[21],B2[21:1]};
 assign sum5 = A3 + {A3[21],A3[21:1]};
 always @(posedge clk or posedge rst)
 begin
    if(rst)
 begin
       bout <= 5'b00000;
       bout_pre <=4'b1101;
end
    else
 begin
       bout_pre <= sum7[21:18] +4'b 1000;
       bout <= bout_pre +5'b01011- 4'b 1000;
end
 end
assign   v_quan = {sum7[21:18],18'h0};
assign   v_back =  ~v_quan + 1'b1;
endmodule
```

Fig. 8: Model of ΣΔ modulator.

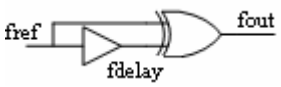


Fig. 9: Frequency doubler.

4 Simulation result

The ideas proposed have been applied to model and simulate the fractional-N PLL frequency synthesizer shown in Fig. 1. The synthesizer is chosen with a reference frequency 19.2MHz, a 1.63GHz-1.69GHz output frequency range, a 0.2mA charge pump current. The jitter of OSC is 35fs, and the phase modulation jitter in VCO is 40fs, and the phase modulation jitter in other driven blocks is about 400fs. The output delay difference of PFD is 0.3ns, and the sink/source current mismatching is 15μA. The SpectreVerilog simulator is used for model verification.

Fig .10 shows the wave of the VCO control voltage. The division ratio is 43.4375 and 43.4375→40.4375. There is only a minute taken to complete the 120μs transient analysis for the PLL loop, but the time is over 24 hours to complete 120μs at the circuit-level, so behavioral models can save much more time. Fig. 11 is the simulated phase noise of the closed loop. The simulation time length is 120mS, and the consumed cpu time is 5 hours or so in a 64bits PC server. Fig.12 is the measured phase noise of the prototype PLL, which can be compared with the plot shown in Fig. 11. Both plots show the noise contributions from the various blocks of the system. The OSC noise contribution is clearly visible in the lower-frequency range. The noise from PFD/CP and divider is dominating in the range of cutoff frequency. The amount of which depends on the level of synchronous jitter exhibited by divider and PFD/CP and the loop bandwidth. The in-band spurs can be observed in the low frequency range in Fig. 11, which is caused by the poor linearity of PFD/CP, especially the dead zone of PFD/CP. The spurs will reduce and even disappear when the dead zone is narrowed or the linearity is improved. The phase noise is dominated by the VCO and ΣΔ modulator in the range that goes from the cutoff frequency up to approximately 10-MHz offset frequency. The noise becomes white beyond the range. The fractional spurs out of the loop bandwidth is mainly caused by the ΣΔ modulator, whereas the reference spur is caused by the frequency doubler which is sensitive to the duty cycle ratio. Due to the limited calculation accuracy, the location and magnitude of the spur is a little different. There is a duty-cycle corrector in the frequency doubler of the prototype PLL, so the reference spur can be suppressed to the noise level. The other parasitic effects have not accounted in the proposed models can be easily added to improve the prediction accuracy, so the proposed behavioral models is helpful to optimization design at the system-level.

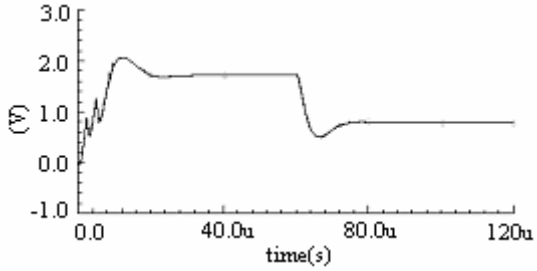


Fig. 10: Closed-loop locked process.

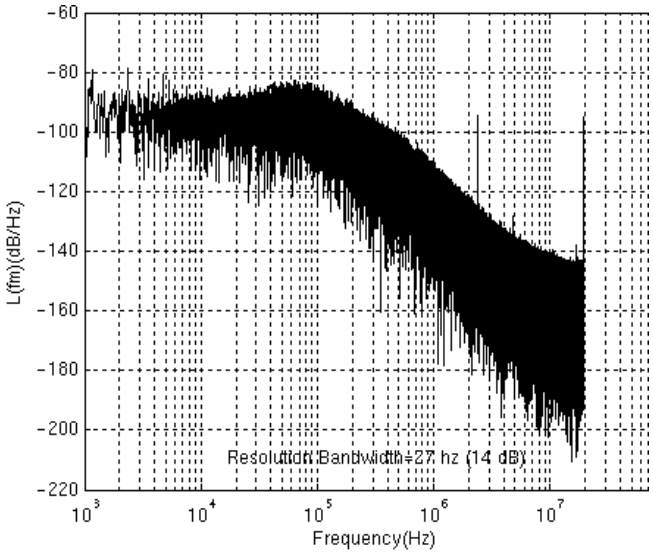


Fig. 11: Phase noise of PLL.

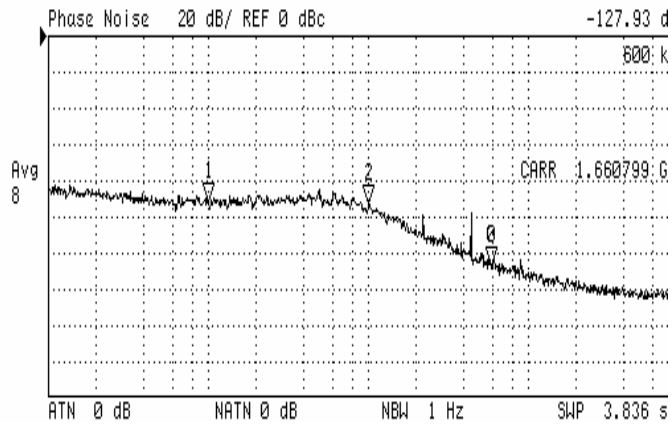


Fig. 12: Measured phase noise of prototype PLL.

5 Conclusion

A set of behavioral voltage-domain verilogA/verilog models are presented in the paper, which are implemented in a three-order ΣΔ fractional-N PLL based frequency synthesizer with a 60 MHz frequency tuning range. The behavioral modeling can provide a great speed-up over transistor simulation, allowing an optimal building block design and giving insight to the key characteristics determining the overall performance by selectively controlling and evaluating the contribution of each noise source and nonideal element. The behavioral models can be calibrated by circuit-level simulation, so the high-level model can accurately analyze the characteristics of dynamic locked process and stable spectral purity. Simulation and measured results verify the flexibility and effectiveness of the behavioral models.

Acknowledgements

This research was partly supported by the National Science Foundation of China (No. 60475018, No. 60372021) and National Key Basic Research and Development Program (No.G2000036508)

References

[1] M. H. Perrot, M. D. Trott, and C.G. Sodini, A modeling approach for Σ-Δ fractional-N frequency synthesizers allowing straightforward noise analysis, *IEEE J. Solid-State Circuits*, 37, 8, 1028-1038, 2002.

[2] Yiwu Tang and Mohammed Ismail, A methodology for fast SPICE simulation of frequency synthesizers, *IEEE Circuits and Devices Mag.*, 16, 4, 10-15, 2000.

[3] K. Kundert, Predicting the phase noise and jitter of PLL-based frequency synthesizer, Available from www.designers-guide.com, May, 2003.

[4] Abhijit Phanse, Ramin Shirani, Behavioral modeling of a phase locked loop, *Southcon/96*, 400-404, 1996.

[5] W. Rhee, B. S. Song and A. Al, A 1.1-GHz CMOS fractional-N frequency synthesizer with a 3-b third-order ΔΣ modulator, *IEEE J. Solid-State Circuits*, 35, 10, 1453-1460, 2000.